An empirical study of the effect of video encoders on Temporal Video Grounding

Ignacio M. De la Jara† Cristian Rodriguez-Opazo† Edison Marrese-Taylor♣ Felipe Bravo-Marquez†
Department of Computer Science, University of Chile, CENIA & IMFD †
Australian Institute for Machine Learning, University of Adelaide‡
National Institute of Advanced Industrial Science and Technology ♣
ignacio.meza@ug.uchile.cl, cristian.rodriguezopazo@adelaide.edu.au
emarrese@weblab.t.u-tokyo.ac.jp, fbravo@dcc.uchile.cl

Abstract

Temporal video grounding is a fundamental task in computer vision, aiming to localize a natural language query in a long, untrimmed video. It has a key role in the scientific community, in part due to the large amount of video generated every day. Although we find extensive work in this task, we note that research remains focused on a small selection of video representations, which may lead to architectural overfitting in the long run. To address this issue, we propose an empirical study to investigate the impact of different video features on a classical architecture. We extract features for three well-known benchmarks, Charades-STA, ActivityNet-Captions and YouCookII, using video encoders based on CNNs, temporal reasoning and transformers. Our results show significant differences in the performance of our model by simply changing the video encoder, while also revealing clear patterns and errors derived from the use of certain features, ultimately indicating potential feature complementarity.

1. Introduction

Understanding and reasoning about long, untrimmed videos is at the core of Computer Vision (CV). As humans have the ability to intuitively identify relevant moments within videos, the task of Temporal Video Grounding (TVG) appears as a fundamental effort in CV, aiming to develop models that recognise and determine temporal boundaries of action instances in videos [10, 12, 28] using natural language queries [7, 13].

As such, work on the TVG task is extensive and includes a wide variety of approaches and techniques. While the original models were mostly suggestion-based, more recent techniques have aimed at predicting the start and end temporal positions directly, or by regressing them from the input video. The recent advent of transformer-based models [32] has also brought new developments, where the integration of pre-training stages or the direct addition of pre-trained vision and speech models has led to significant performance improvements.

Despite the large amount of prior work in the TVG task, we find that the role of the video representation has not been consistently investigated so far. Specifically, we observe that prior work has relied on features derived from action classification models such as C3D [30] or I3D [2], with alternatives remaining relatively unexplored. We speculate that this selection bias may lead to model designs that overfit or exploit spurious patterns in these features, without generalising in the long run. We suggest that the increased complexity of recent approaches in search of improved performance can be seen as indirect evidence of this point.

In light of this issue, in this paper we propose a comprehensive empirical study to shed light on the role of video representation in the TVG task. We consider three relevant benchmark datasets, Charades-STA, ActivityNet Captions, and YouCookII, and design a comprehensive framework that allows us to isolate the effect of different video representations by introducing minimal changes in the model architecture. We use a wide variety of pre-trained models to extract video representations, including more than 10 different types of models, resulting in more than 30 sets of extracted features for our data. These include, but are not limited to, well-known CNN and Transformer-based action classifiers. We release features to encourage research in this area.

Our results show that changes in the video representation can lead to substantial performance improvements by keeping the models as is, allowing a “classical” approach [25] to perform better by a large margin by simply changing the video encoder, findings that are consistent with similar observations recently made in the context of query representations [16]. Our experiments reveal clear pitfalls in the use of certain features and uncover complementarity between features, which could lead to further performance improvements if exploited.

1https://github.com/Mezosky/VideoFeatures_TVG
2. Related Work

**Action Classification** The task of identifying the action being performed in a video is the cornerstone of current video understanding pipelines. It is essential for the task of temporal video grounding, as it is the current way of encoding videos. Current methods are usually trained via supervised learning, using datasets such as Something-Something [11], Kinetics [2], among others, which contain short video clips of a single action.

Different methods have been proposed to solve the action classification task, the most recent approach based on neural networks could be divided into CNN-based [2, 5, 6, 31], temporal reasoning [19, 33] and transformers [4, 18]. In the case of CNN-based, the methods exploit the ability of CNNs to capture spatial information in images and add various techniques to extract the temporal information from the video. [31] proposed a convolutional block, which is the combination of 2D convolutions followed by a 1D convolution. Such blocks approximate the behaviour of 3D convolutions and allow to capture the temporality from the videos. Knowing that the information in a video is very redundant and not symmetric, [6] proposed to use a slow and a fast track, aiming to capture the spatial and temporal information respectively. Temporal reasoning classifiers aim to capture the temporal information between features extracted from frames using techniques that exploit latent information in videos, such as redundancy (sampling), summarisation (aggregation), and ordering (ranking). Limin et al. [33] use a sampling technique to select features that, after passing through an aggregation mechanism, effectively represent the video. Aiming to create a better representation of the videos, Lin et al. [19] proposed a shift module that applies a shift in part of the temporal channel of the features to exchange information between frames. The advent of transformer-based models has also led to new developments in this task [1, 4]. However, their success in this task is due to their flexibility in capturing spatial information and their inherent ability to model sequential information. [4] presents a multiscale feature hierarchy that combines low-level spatial information processed at early layers with more complex high-level features that capture temporal information processed at deeper layers.

**Temporal Video Grounding** The task that concerns this work, where we find mainly two types of approaches. On the one hand, we find proposal-based models that, given a query, return a set of candidates that can later be ranked [3, 8, 20, 36, 38]. Recent approaches include both transformer-based models and contrastive losses [24, 27, 35, 37, 39]. On the other hand, we find proposal-free models that try to predict the start and end locations directly from the video span [9, 25].

3. Experimental Framework

**Model** At the core of our experimental framework lies the TVG model to be used as a pivot for testing video features. Although there are a variety of approaches to our task, we find that not all of them are suitable for our purposes, as replacing the video representations can in many cases lead to substantial changes in the model architecture and/or training. After careful consideration, we selected the proposal-free approach proposed by Rodriguez et al. [25] (TMLGA), which can be summarized in three parts: 1) a text encoder based on GloVe [23] embeddings on top of an LSTM [14], 2) a video encoder, which we discuss in detail below, and, 3) a localization module that combines information from both modalities using a dynamic filter [15] and predicts the start and end points of the segment. We consider this model to be a “classic” approach, as it has been widely adopted as a baseline by recent work, allowing for meaningful comparisons with a wide variety of approaches. We also note that TMLGA relies on a rather simple approach to query representation, which we believe allows for a cleaner result in terms of the contribution of the language encoder. Finally, we point out that TMLGA has an official implementation that has been publicly released, which we believe is essential to facilitate the replication of the original results and ultimately ensure the reproducibility of our experiments.

**Video Representation** Following previous work, our pivot TMLGA model uses pre-trained action classification models to obtain a sequence of vectors to represent a given video. Critically, these input vectors are just passed through one projection layer before being fed into the localization module, which ensures that the amount of changes introduced to the model is minimized for our experiments. Although the original implementation relies on I3D [2] we instead consider a wide selection of pre-trained models for feature extraction, which we propose to group into three main categories. First, we test CNN-based video classification approaches, including C2D [30], I3D, SlowFast [6], X3D [5] and Non-Local variations in one classifier [34]. For these models, we modify the classification head of the last ResNet or ConvNet used, extracting the representation generated before dropout, final projections and classification layer of the network. Secondly, we look at Transformer-based action classification models, including MViT [4], MViT2 [18], and Rev-MViT [21]. Similarly to the CNN-based case, for these models we represent the video using the vectors of the last layer of the transformer before the linear classifier. Finally, we study temporal reasoning models such as TSM [19], which provide a different angle in tackling the action classification task. In this model, the video data is divided into segments, each containing 8 buckets, where each bucket consists of 8 consecutive frames. This segmentation approach creates smaller
video subsequences, allowing the temporal network to effectively permute frames and capture their temporal dependencies. Similarly to the other cases, the last projections of the temporal network are discarded to obtain the video feature representation.

Following Rodriguez et al. [25], the video encoding procedure is done offline, performing rescaling to 320x240 when necessary, and sampling at 25 fps. Our implementation is based on decord\(^2\), pyslowfast\(^3\) and the official code release for TSM [19]\(^4\). Each video is sequentially traversed in buckets of a size given by the frame rate of the backbone model. In cases where the total number of extracted frames is not divisible by this number, we fill the last bucket by repeating the last frame.

**Datasets** We consider three well-known challenging benchmarks for the task. Charades-STA [7, 29], ActivityNet-Captions [17], and YouCookII [40, 41]. We use the pre-defined train and test sets. Each dataset with its own challenges, from more ambiguous spatio-temporal information to very long videos.

**Evaluation** To assess the effectiveness of our extracted features, we conduct two kinds of analysis. First, a quantitative one based on metrics proposed in [7], namely the recall with thresholds (\(\alpha\)) of 0.5 and 0.7 for the temporal intersection over union (tIoU). Secondly, we provide an in-depth qualitative analysis of the model predictions, focusing on the biases presented when training with different features.

### 4. Results

**Quantitative Analysis** Table 1 summarizes the results of our experiments, showing the performance of the TMLGA model trained with different features. As can be seen, in the case of the Charades-STA dataset, it is clear that the Transformer-based features give promising results, while our 1D features still perform poorly. This is in contrast to the original results, where the 1D features were only able to deliver 2.45 points over our results. We attribute this difference to the pre-training dataset, as the original features were pre-trained on Charades, which may have led to overfitting. The proposed features, when pre-trained on the K400 dataset, show competitive performance compared to the original results. This allows for unbiased selection and avoids the need for time-consuming training on the task-specific dataset, thereby accelerating the development of architectures for localization tasks.

On ActivityNet, we note two interesting observations. First, we find that some CNN-based features can outperform more recent Transformer-based models, with differences of up to 7 points in the 0.7 band. We also notice that the overall performance of the features shows less variation, suggesting that the query representation may be particularly relevant. This is consistent with the data showing that ActivityNet has relatively more complex queries in terms of vocabulary size (748 vs. 9,744 tokens) and length (7.2 vs. 13.48 tokens per query).

The results on YouCookII show that this dataset remains the most visually challenging, which we suspect is partly due to the length of the videos, with an average of around 5 minutes. On this dataset, Transformer-based models again achieve the best performance, with results similar to complex architectures such as DORi [26], which make explicit use of spatial information.

Overall, our results highlight the importance of careful feature selection in the TVG task, as we see that a simple change can lead to significant performance gains of up to 5 points. Ultimately, we see how selecting features without a thorough understanding of their impact on the dataset can lead to stagnation in development and suboptimal performance. Further research is suggested to investigate the effect of different features and to explore their applicability to other video and speech tasks.

**Qualitative Analysis** To visualize the output of the model when given different features, we plot the 0-1 normalized predicted time intervals for each query, similar to [22]. We generate one plot for correct intervals and another for incorrect predictions, based on a given \(\alpha\) of 0.7 tIoU band.

As shown in Figure 1, we see that in the case of Charades-STA, different features produce two points of correct predictions, represented by the lower-left and upper-right accumulation points. We see that in many cases, the

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Encoder</td>
<td>Pretrain</td>
<td>0.5 0.7</td>
<td>0.5 0.7</td>
</tr>
<tr>
<td>MVIT</td>
<td>8 K400</td>
<td>50.27 31.32</td>
<td>27.78 15.21</td>
</tr>
<tr>
<td>MVIT-v2</td>
<td>16 K400</td>
<td>48.17 30.27</td>
<td>25.40 13.95</td>
</tr>
<tr>
<td>Rev-MViT</td>
<td>16 K400</td>
<td>49.89 32.31</td>
<td>23.71 13.00</td>
</tr>
<tr>
<td>X3D M</td>
<td>16 K400</td>
<td>43.60 27.96</td>
<td>22.97 12.80</td>
</tr>
<tr>
<td>X3D S</td>
<td>13 K400</td>
<td>41.37 25.16</td>
<td>23.00 12.60</td>
</tr>
<tr>
<td>SlowFast</td>
<td>8 K400</td>
<td>38.82 24.22</td>
<td>21.39 11.57</td>
</tr>
<tr>
<td>SlowFast 16</td>
<td>SS V.2</td>
<td>39.76 23.90</td>
<td>21.16 11.05</td>
</tr>
<tr>
<td>SlowOnly</td>
<td>8 K400</td>
<td>36.16 20.19</td>
<td>11.31 6.24</td>
</tr>
<tr>
<td>1D NLN</td>
<td>8 K400</td>
<td>38.63 24.33</td>
<td>20.76 11.60</td>
</tr>
<tr>
<td>C2D</td>
<td>8 K400</td>
<td>02.63 00.97</td>
<td>08.85 03.92</td>
</tr>
<tr>
<td>TSM</td>
<td>8 K400</td>
<td>48.35 22.70</td>
<td>13.29 13.29</td>
</tr>
<tr>
<td>Mean</td>
<td>-</td>
<td>39.51 23.98</td>
<td>21.01 11.38</td>
</tr>
<tr>
<td>STD</td>
<td>-</td>
<td>13.41 06.65</td>
<td>06.76 03.37</td>
</tr>
</tbody>
</table>

Table 1: Summary of our experimental results, where \(f\) denotes the frame rate used to extract features, while K400 and SS V.2 stand for the Kinetics-400 and Something-Something V.2 datasets, respectively. In the table, the best results for each dataset are indicated in **bold**, while the worst results are underlined.

---

\(^2\)https://github.com/dmlc/decord  
\(^3\)https://github.com/facebookresearch/SlowFast  
\(^4\)https://github.com/mit-han-lab/temporal-shift-module#pretrained-models
5. Conclusions

In this paper, we have conducted a comprehensive analysis of video features in the context of the video grounding task. Our research shows that using features with pre-trained encoders from datasets different from the target application can produce similar results to using a fine-tuned encoder specific to the application dataset. This approach effectively mitigates potential network overfitting and avoids biases arising from assumptions about task proficiency. Our investigation has also revealed exclusive predictive patterns among video features, suggesting the potential for improved predictions through their integration into a unified network. Furthermore, we have shown that varying the features can have a significant impact on localisation, leading to performance improvements comparable to more complex networks that explicitly consider spatial components in their inputs. These results provide valuable insights into how the spectrum of feature diversity and encoder pre-training can be exploited to advance the field of video analysis and improve the field of temporal localisation tasks.

Looking ahead, a notable avenue for future work lies in the application of these features within modern architectures. In addition, it is anticipated that the inherent orthogonality of these features will be exploited to build more robust frameworks tailored to the task of temporal localization.

Acknowledgements

Ignacio Meza and Felipe Bravo-Marquez were supported by ANID Millennium Science Initiative Program Code ICN17_002 and the National Center for Artificial Intelligence CENIA FB210017, Basal ANID.
References


[21] Karttikeya Mangalam, Haoqi Fan, Yanghao Li, Chao-Yuan Wu, Bo Xiong, Christoph Feichtenhofer, and Jitendra Malik. Reversible vision transformers, 2023. 2


[38] Songyang Zhang, Houwen Peng, Jianlong Fu, Yijun Lu, and Jiebo Luo. Multi-Scale 2D Temporal Adjacency Networks for Moment Localization with Natural Language. IEEE Transactions on Pattern Analysis and Machine Intelligence, pages 1–1, 2021. 2

